On the use of phone-gram units in recurrent neural networks for language identification

Christian Salamea[1,2], Luis Fernando D'Haro[3], Ricardo de Córdoba[2], Rubén San-Segundo[2]

[1]Universidad Politécnica Salesiana del Ecuador
Calle Vieja 12-30 y Elia Liut, Casilla 26, Cuenca, Ecuador
csalamea@ups.edu.ec

[2] Speech Technology Group. Dpto. de Ing. Electrónica. Universidad Politécnica de Madrid.
Ciudad Universitaria s/n 28040, Madrid, Spain.
cordoba@die.upm.es, lapiz@die.upm.es

[3]Human Language Technology Institute for Infocomm Reseach (A-STAR)
1 Fusionopolis Way, #21-01 Singapore 138632
luisdhe@i2r.a-star.edu.sg

Abstract

In this paper we present our results on using RNN-based LM scores trained on different phone-gram orders and using different phonetic ASR recognizers. In order to avoid data sparseness problems and to reduce the vocabulary of all possible n-gram combinations, a K-means clustering procedure was performed using phone-vector embeddings as a pre-processing step. Additional experiments to optimize the amount of classes, batch-size, hidden neurons, state-unfolding, are also presented. We have worked with the KALAKA-3 database for the plenty-closed condition [1]. Thanks to our clustering technique and the combination of high level phone-grams, our phonotactic system performs ~13% better than the unigram-based RNNLM system. Also, the obtained RNNLM scores are calibrated and fused with other scores from an acoustic-based i-vector system and a traditional PPRLM system. This fusion provides additional improvements showing that they provide complementary information to the LID system.

1. Introduction

Automatic spoken language identification (LID) is the process of identifying the actual language of a sample of speech using a known set of trained language models. Currently, there are two main methods to achieve this goal: the first method uses acoustic features extracted from the speech signal, while the second method uses as features the sequences of transcribed text (typically phones) obtained using an automatic speech recognition system (ASR). In general, acoustic-based systems achieve the best performances, although the combination with a phonetic-based system provides a higher accuracy when both kind of features/scores are fused [2], [3]. This paper is mainly focused on improving the results of a phonetic-based system using the same structure of a PPRLM-based system [4]. In this case, our system is made of two components: a) the "Front-End" where several parallel phonetic recognizers are used to generate the sequence of phonemes for a given speech audio file, and b) the "Back-End" where the sequence of phonemes are used to train different language models (one for each language to recognize and phone recognizer) whose scores are finally compared among them in order to select as recognized language the language corresponding to the model which generates the lowest perplexity for the given test utterance.

Since the scores produced by each language model are biased [4], due to the different number of phone units used by each ASR and the amount of training data, a calibration step is required before the classifier. Besides, the combination of scores from different levels and sources of information (e.g. acoustic features, higher n-gram orders) provide complementary information, so fusion techniques are also applied to get better performances.

In our system, we will combine traditional language models [5] with more recent recurrent-based language models [6] trained with the output of three different ASR phone recognizers, and fused with an acoustic i-vector based system. Results are presented for the plenty-closed condition of the KALAKA-3 database provided during the Albayzin LRE 2012 evaluation [1]. The audio used in this database was very challenging since it was collected from YouTube videos, with different length durations, channel conditions, number of speakers, and several kinds of noises. All audio files were sampled at 16 KHz. During the evaluation, four different conditions were proposed depending on the languages to recognize, the availability of training data (plenty or empty), and the possibility of recognizing out-of-set languages or not (open vs closed). This paper shows results only on the main condition (plenty-closed), where the target languages were: Spanish, Catalan, Basque, Galician, Portuguese, and English; the total number of files for the train set was 5115, 458 for development, and 941 for eval.

This paper is organized as follows: In section 2, the phone-gram units concept and the system components are explained. In section 3, the different phone-based RNN-LM models and acoustic model used in this work are described. Then, in section 4, we present and discuss our results. Finally, in section 5, we present our conclusions and future work.

2. System Description

2.1. The concept of phone-gram units

Language models used in PPRLM-based systems can be trained using different algorithms. For instance, in 2001 [5] proposed maximum entropy models, while [7] proposed using neural network models, and Mikolov [6], in 2010, successfully proposed using recurrent neural networks. In Mikolov's model,

the state layer information is stored in the form of a multidimensional vector representation of two or more temporal states in the neural network, and then it is concatenated with the elements of the input layer of the neural network. The multidimensional vector representation is obtained applying a learning process which is based on the application of the Back-Propagation Through Time algorithm (BPTT) [8], [9] where the weights used by the RNN are calculated considering the temporal information obtained from the past states of the RNN.

The model proposed by Mikolov [6] is designed to model phonetic structures at a word level, which is clearly efficient. However, the problem gets worse when the phonetic structures are phonemes, as the model needs a state layer which is three times bigger than the layer used with words [10] to obtain similar results. There are two important drawbacks for phonemes. First, there is an important increase of the computational cost, and second, the systems can be easily over-trained [11].

We have considered the model proposed by Mikolov adapted to phonemes as our baseline. Then, we will show how the n-gram phones concept applied to the phonetic sequences improves the baseline performance. We have used a 1-N codification for the phonetic units, being N the total number of phonemes (our vocabulary).

As we know the relevance of the phoneme context in a LID task based on phonotactic information, with this approach we look forward to improve the RNNs behavior by incorporating contextual information in the inputs and, for that objective, we propose the concatenation of n-adjacent phonemes in a structure called phone-gram unit. As we can see in Figure 1, for diphones and triphones the new sequence includes contextual information in a more explicit way in the RNN, both in the state layer and in the projection layer where usually the temporal information is stored.

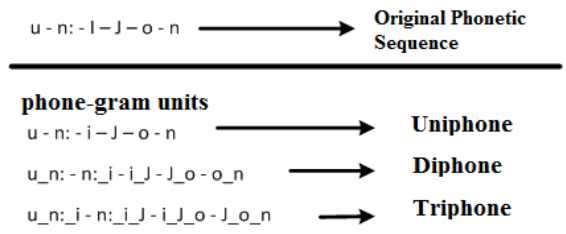

Figure 1*: Generation of the phone-gram units from a phoneme sequence.*

With this approach, we expect to improve the performance of the RNNLMs that use only phoneme sequences in the training process. The generation of high-order phone-grams has the implicit drawback of the increase in vocabulary size. For instance, for the database used for this work, [1] and using one of the phoneme recognizers from Brno University [12], specifically the Hungarian recognizer, the phone-grams of 1 element or uniphones have a vocabulary of 61 elements (obviously, the same number obtained with the original model that uses phonemes), the phone-grams of order 2 or diphones have a vocabulary of 1938 elements and the triphones have a vocabulary of 28097 elements. All of them are the units observed considering Spanish utterances from the training data with the Hungarian recognizer.

2.2. Phoneme Recognizers

The phoneme recognizer, the main component of the "Front-End" in the PPRLM structure, is based on the system designed by the Brno University (BUT) [12], which uses monophone three state HMMs. There are 3 HMMs (Hungarian, Russian and Czech) with 61 different types of phonemes, 46, and 52 respectively.

2.3. RNNLM-P applied to Language Identification

As mentioned in previous sections, a PPRLM architecture has been used in this work. For each Phonetic recognizer [12] used in the Front-End a phoneme sequence is obtained. These sequences are used to generate the new phone-grams sequences and to train the models for each language in a supervised way. For each phone-gram sequence from a evaluation utterance, an entropy metric is obtained for each language,. Then, these scores are calibrated and fused [13] to obtain a global score. Finally, the complete system is evaluated using the Cavg metric, that takes into account the "False Acceptance" and the "False Rejection" errors [14].

2.4. Phonetic Vector Representation

In our approach, neural embeddings are vector representations of the input elements used for a LID task.

This way, the objective of the neural embedding models is to predict the phonetic unit that is going to appear next according to the context where the unit is included. From several models that have been proposed, two of them are the most used: the Skip-Gram Model and the C-Bow Model.

Using a Skip-Gram model, a phonetic vector representation is obtained, which is useful to predict the context words in a sentence or document.

The model definition normally used to train the embeddings is focused at the word-level, so it is based on the hypothesis that words in similar contexts tend to have similar meanings. The distance between words with a similar syntax and semantics tends to be small, while the distance between words with a different syntax and semantics tends to be higher.

In our case, where we work at the phone level. We are looking forward to finding co-ocurrence of phonemes and sequences of phonemes that tend to happen in similar contexts for a specific language. This way, we expect to improve the results compared with the results obtained when only uniphone sequences are used. Since vector representations have been usually treated at a word level and, on the other side, our study focuses on phonetic units that we have called "phone-grams", and their use in the continuous space has been called Phonetic Vector Representation.

In general, the models are characterized by the relationship between the phone input, its context and the context representation [15]. The representation of the context used to calculate the conditional probability is defined by the Skip-Gram model or the CBow model. We have selected Skip-Gram as we obtained better results in initial experiments.

2.4.1. Skip-Gram model

The training objective of the Skip-Gram model is to learn the phonetic vector representation of a phone-gram that could be "good" to predict its context in the same sentence. The context of the sentence is represented by one of the phone-grams in the window *v* that contains the context.

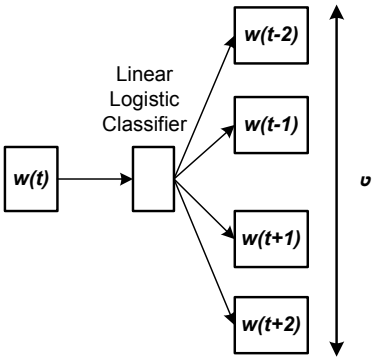

Figure 2*: Skip-Gram model.*

The model is forced to predict random sampled phone-grams from a context defined in a window *v* [16]. This means that, in each iteration of gradient descent (learning process of the network), a section of the phonetic sequence of window *v* is sampled, and just there, the phone-grams in that window are randomly sampled, this way a classification phase between the phone-grams and the context in that window is created.

The phone-gram in time *t* is used as an input to a linear-logarithmic classifier with a projection layer and predicts the occurrence probability of a phone-gram in the same *t* given other phone-grams randomly selected from the context window, either before or after the phone-grams that appeared in time *t* [17] (Figure 2).

2.4.2. Vocabulary reduction using phonetic vector representation

High order phone-grams imply an increase of the number of phonetic units and, so, their dispersion and the appearance of units with a low number of examples in the training database, and so they suffer from an unreliable estimation. In this work, we have used the vector representation of phone-grams as the input in a vocabulary reduction process using the k-means algorithm.

Our focus is to eliminate from the vocabulary the less representative phone-grams of a language, and with the resultant vocabulary to train the language models using the recurrent neural network. We considerate that vocabulary reduction will decrease the scattering of training information, and we could obtain more robust language models. The system can be seen in Figure 3.

The k-means algorithm is an unsupervised classification algorithm that is useful to group scattered data or observations in different groups usually called “classes”. The method used to define the classes for each phone-gram in the vocabulary is the following: first, the embeddings must be generated, as the distance between the units will be based on that. After that, the final number of classes is defined, which will be the number of centroids used by the k-means algorithm. Finally, for each phone-gram, the distance between its embedding and each centroid is computed, choosing the closest as the unit class..

2.5. Systems Fusion

The objective of the fusion is to make use of information obtained from different modules to extract the best contribution from each one and obtain a general improvement in the results [18]. Usually, this type of information is related to probability values or scores and methods of linear or logarithmic combinations that assign weights to every model implicated in the fusion. Information from the three phoneme recognizers has been used in this work to generate the decision scores and subsequent fusion.

3. System Configuration

3.1. In relation to the neural network

The right performance of the neural networks in the generation of the language models is conditioned to an appropriate configuration of their parameters. Among the most important ones we should mention:

1. The Number of neurons in the state layer (NNE).
2. Number of classes (NCS). Phone-grams are grouped in the output layer in a factorization process [19], [20], [21], where the phone-gram probability is the probability of the class multiplied by the probability of the phone-gram given the class. A high NCS value speeds up the RNN training but the final language model is less accurate.
3. Number of the state layers (MEM) corresponding to the previous times. With this parameter, previous context information is taken into account.
4. Number of times the network output values are processed before upgrading the network weights (ORD). This parameter is not especially relevant in comparison with the other ones.

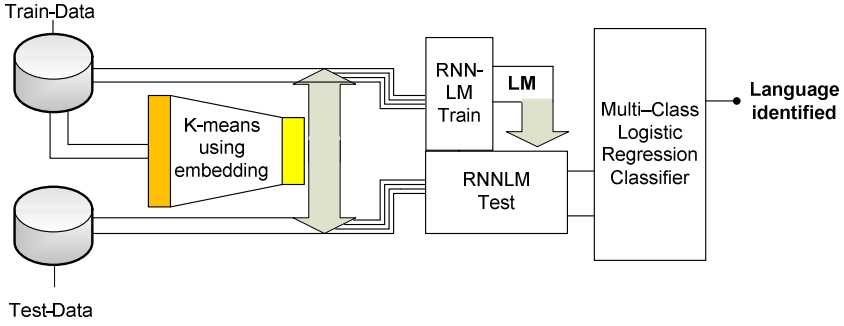

Figure 3*: LID system used to reduce the vocabulary size of phone-grams.*

To evaluate the behavior of the RNNLMs being generated using phone-grams (RNNLM-P), we must be aware of the vocabulary sizes obtained. For instance, the vocabulary sizes for uni, di and triphones in the case of the Spanish training set for each of the phonetic recognizers are shown in Table 2:

	Number of phone-grams		
Phone-gram	Russian	Hungarian	Czech
Uniphone	52	61	46
Diphone	1,876	1,938	1,572
Triphone	29,822	28,097	25,874

Table 2: *Number of phone-gram units found in the Spanish train set for each phone recognizer*

For the other languages, we obtain similar figures, so we can use Table 2 as a reference. We can see that, as could be expected, the vocabulary increases drastically as the phone-gram order does, with the dispersion problems already mentioned in Section 2.4.2. To deal with it, the factorization of the output layer and the number of the neurons in the state layer (NNE) have been modified.

3.2. In relation to the embedding modeling

To select the optimum model for the embeddings (either Skip-Gram or CBow), we have trained an i-vector system using as inputs the trained embeddings. Each phone-gram unit in the phonetic sequences used to train the i-vectors was replaced by its respective embedding sequence. These sequences have been used as feature vectors to train a total variability matrix and an Universal Background Model (UBM), which are used to obtain the i-vectors of each evaluation utterance (the method is similar to the method used with the acoustic parameters). The resultant i-vectors are used to train a multiclass logistical regression classifier where the scores are calibrated and fused. The obtained Cavg [14] was used to determinate the best embedding model.

3.3. Acoustic i-vector-based system using MFCCs

From each speech evaluation utterance present in a voice file, 12 coefficients MFCCs [22] that include C0 are extracted for each frame. The silent and noise segments of the acoustic signal have been removed using a Voice Activity Detector. To reduce the noise perturbation, a RASTA filter has been used together with a cepstral mean and variance normalization (CMNV). Frames of speech separated 10 ms were projected in a feature vector of 56 dimensions, generated from the concatenation of the SDC parameters using the 7-1-3-7 configuration. Feature vectors are used to training the total variability matrix, from which the i-vectors of dimension 400 with 512 Gaussians are extracted (optimal configuration).

4. Results

Based on the parameters defined in the previous section, a particular analysis of each phone-gram order was performed to determine the optimal configuration in each case. In all cases, the results in the tables correspond to the fusion of the three phonetic recognizers (Russian, Hungarian, and Czech).

4.1. Language models based on RNNs for uniphones

For this condition, NCS=1 has been chosen because of the small vocabulary size (see Table 2), as the factorization of the output layer is not necessary. In preliminary tests, the optimal number of neurons in the state layer has been determined as NNE=250. In relation to the previous times considered by the RNN to include past information (parameter MEM), we have experimented with several values, finding an optimum for MEM=3. So, we consider this value as the optimal and the RNN configuration is defined by NNE=250, MEM=3, NCS=1 and ORD=5. We can see the results in Table 3.

4.2. Language models based on RNNs for diphones

For diphones the vocabulary size is around 2,000. Therefore, the factorization of the output layer is necessary and finding the optimum number of classes in the output layer (NCS) is a must. We have evaluated the RNN performance for 3 different values for NCS (1, 30, and 60) and varying the number of the neurons in the state layer. We found that the performance of the RNN with diphones for NCS=30 is better than the result obtained with uniphones.

This way, in our database the optimal configuration using diphones is NCS=30, NNE=100, MEM=20 and ORD=5.

4.2.1. Vocabulary reduction of diphones using vector representations

As we described in Section 2.4.2, we have used vector representations to replace the diphones with few appearances in the training set by the nearest one as defined by the clusters obtained by the k-means algorithm. With the new phone-gram sequences, the RNN training process was performed with the same configuration used previously.

Applying this technique, we have improved the recognition results up to 7.34% relative compared to the LID system that uses diphones with no vocabulary reduction (see Table 3). On the other side, there is an improvement of 10.3% relative compared to the system with uniphones.

	Cavg	
Phone-grams	Abs	Improve%
Uniphone	12.81	
Diphone	12.40	3.2
Diphone + vocab_reduct	11.49	10.3

Table 3: *Behavior of Cavg for Uniphone, Diphone and Diphone + vocabulary reduction*

4.3. Language models based on RNNs for triphones

The vocabulary size for triphones is around 28k (see Table 2) in our database. The optimal configuration of the RNN starts with the following values: NCS=100, MEM=3, ORD=5 and NCS=300, and varying the number of the neurons in the state layer NNE. We have determined that using 200 neurons (NNE) is the optimum. Next, we evaluated the quantity of the time memory required to the RNN (MEM parameter). The results can be seen in Figure 4:

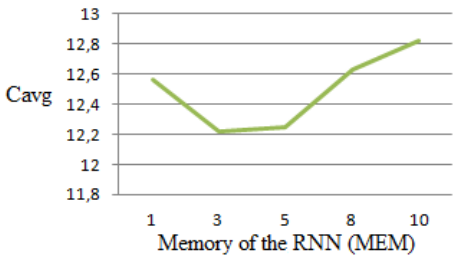

Figure 4: *Optimal Memory factor for the RNN using triphones.*

As we can see in Figure 4, although the optimum is obtained with 3 to 5 memory states, the performance of the RNN gets worse when this value increases. Analyzing the reasons of this behavior, we propose the following hypothesis: the intra and between classes scattering for triphones and the reduced number of appearances of a lot of them, complicate the RNN training. The hypothesis is based on the fact that we do not observe this behavior for uniphones nor diphones, where the scattering is very low. We also observed that the RNN applies implicitly a penalty on the score for unseen units in the training set. This can be seen in the log-likelihood obtained when we compare the RNNLM with a classical language model of phonemes (SRI-LM) in two evaluation triphones. In both cases, these triphones did not appear in the training phase. (Table 4).

Triphone	RNNLM	SRI-LM
i_i:_x	-4.7917	-2.1383
m_i_u:	-4.6339	-2.2294

Table 4: *Log-likelihood for two triphones obtained with RNNLMs and a classical LM*

As we can see in Table 4, the penalty applied by the RNNs to the log-likelihood is higher than the penalty applied by the classical language models [5], and for that reason, we decided to apply a threshold so that triphones that do not appear in training do not deteriorate the perplexity generated by the rest of the triphones present in the evaluation sentence.

This threshold is applied as follows. First, the average of the occurrence probability in the sentence is calculated from each triphone in the phone sequence. Second, this average is divided by a numeric constant *k* whose value is selected empirically to obtain the final value. This threshold value is assigned to all unseen triphones. Equation 1 reflects this procedure:

$$Umb = \frac{1}{Nk}\sum_{i=1}^{N} p(phi_t|phj_{t-1}) \quad (1)$$

Where $p(phi_t|phj_{t-1})$ is the occurrence probability of a triphone *phi* given the triphone phj that appears at time t-1. *k* is the numeric constant. *N* is the number of triphones in the evaluation file.

To find the optimum k, we carried out experiments varying k between 1 and 10, keeping the configuration parameters NNE=200, NCS=300 and MEM=8. We considered a non optimal value for MEM because we believe that the effect of the threshold could be more evident with a worse MEM value. The results can be seen in Figure 5.

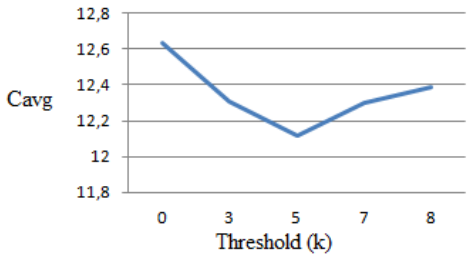

Figure 5: *Modification of k in the threshold formula*

In Figure 5 we can see that the best threshold value is *k=5*. Next, we continued the evaluation with fixed values for NNE=200, NCS=300 and a variable MEM. In Figure 6 we compare the results without the threshold (taken from Figure 4) and with the threshold. Now, the optimum is for MEM=5. So, we will use this configuration as the optimal for triphones.

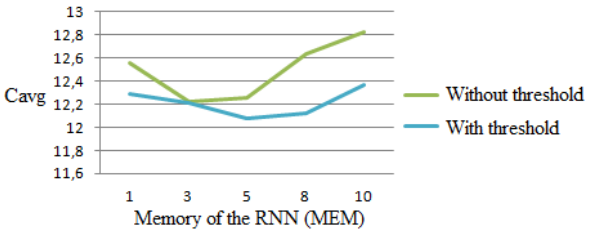

Figure *6:* Comparison of the triphone system with and without the threshold.

Similar tests have been performed using diphones, but no improvements have been observed, which could be expected as the number of unseen diphones is extremely low, so the threshold is very rarely applied.

4.4. Comparison of uni-, di- and triphones

We will now compare the results of the proposed technique, RNNLM-P, for uni, di, and triphones. The results can be seen in Figure 7, where it is obvious that the best recognition rate is obtained for triphones, although the effect of increasing the memory states in the RNN is useful until a value of MEM=5 where a relative improvement of 7.7% is obtained compared to the optimum for uniphones.

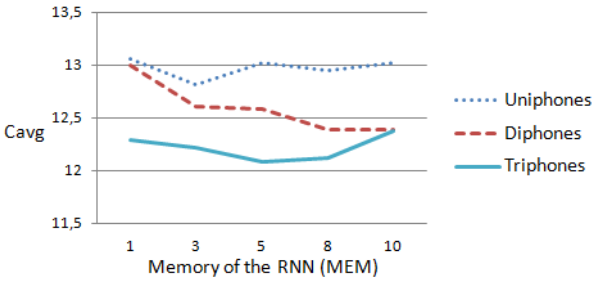

Figure *7:* Best results for uniphone, diphone, and triphones

4.5. Fusion of the three phone-gram orders

The next step is to fuse these systems. Results in Table 5 have been obtained for the following configurations, which correspond to the best topology for each phone-gram order:

- Uniphone: NNE=250, MEM=3, NCS=1 and ORD=5
- Diphone: NNE=100, MEM=20, NCS=30 and ORD=5
- Triphone: NNE=200, MEM=5, NCS=300 and ORD=5

	Cavg	
RNNLM-P Trial	Abs	Improve%
Uni-gram	12.81	
Diphone	11.49	10.3
Triphone	11.92	6.9
Fusion	10.87	15.1

Table 5: *Fusion of uni, di and triphones*

In Table 6 we compare the results obtained by the RNNLM-P system proposed in this paper with the PPRLM and MFCCs systems. The RNNLM-P provides a 6.1% relative improvement over the PPRLM system.

System	Abs
MFCCs	7.60
PPRLM	11.57
RNNLM-P	10.87

Table 6: *Best results for all individual systems*

In the case of the PPRLM system, the language models have been obtained applying the Witten-Bell technique to smooth the model.

In Table 7, the final global result for all fusions are shown, combining the three systems, RNNLM-P, PPRLM and MFCCs. Improvements are also computed in relation to the PPRLM system There are relevant improvements in all cases, with contributions to both PPRLM and MFCC systems. We can also see that the combination of MFCC with the proposed technique is better than with PPRLM. In any case, the combination of the three systems further improves the results.

	Cavg	
System	Abs	Imp%
RNNLM-P+PPRLM	10.51	9.2
PPRLM+MFCCs	5.10	32.9
RNNLM-P+MFCCs	5.04	33.7
RNNLM-P+PPRLM+MFCCs	4.80	36.8

Table 7: *Best results for all fused systems*

5. Conclusions and future work

The proposed technique, based on using phone-gram units for the LID task provides better results than the original technique, based on using characters for the language models generation. Also, the system benefit from the fusion of phone-gram orders 1-2-3 with a 13% relative improvement.

We have also presented the best parameter configurations of the RNNLM for all phone-gram orders.

Finally, the fusion of RNNLM-P with other language recognition systems, namely an acoustic based system and a PPRLM system provides improvements in all cases, up to 36.8%. So, we can conclude that the phonetic vector representation can be successfully used for the LID task.

As future work, we expect to improve the performance of the RNNLM-P thanks to the inclusion of discriminative information obtained from rankings of the most discriminative phonemes between languages. In relation to the phonetic vector representation for the LID task, we will evaluate other clustering techniques to the reduction of the vocabulary, as well as dealing with the OOVs present in test sentences.

6. Acknowledgements

This work has been supported by ASLP-MULÁN (TIN2014-54288-C4-1-R), NAVEGABLE (MICINN, DPI2014-53525-C3-2-R), MA2VICMR (Comunidad Autónoma de Madrid, S2009/TIC-1542), SENESCYT, and the Universidad Politécnica Salesiana de Ecuador.

7. References

[1] L.J. Rodriguez-Fuentes, N. Brümmer, M. Penagarikano, A. Varona, G. Bordel, and M. Diez, "The albayzin 2012 language recognition evaluation." in Interspeech, 2013, pp 1497 - 1501.

[2] C. R. Salamea, L. F. D'Haro, R. Córdoba, and M. Á. Caraballo, "Incorporation of discriminative n-grams to improve a phonotactic language recognizer based on i-vectors" Procesamiento del Lenguaje Natural, no. 51, pp. 145 – 152, 2013.

[3] J. Gonzalez-Dominguez, I. Lopez-Moreno, H. Sak, J. Gonzalez-Rodriguez, and P. J. Moreno, "Automatic language identification using Long Short-Term Memory recurrent neural networks," in Proc. Inter-speech, 2014.

[4] M. A. Zissman et al., "Comparison of four approaches to automatic language identification of telephone speech," IEEE Transactions on Speech and Audio Processing, vol. 4, no. 1, p. 31, 1996.

[5] S. F. Chen and J. Goodman, "An empirical study of smoothing tech-niques for language modeling," in Proceedings of the 34th annual meeting on Association for Computational Linguistics. Association for Computational Linguistics, 1996, pp. 310 – 318.

[6] T. Mikolov, M. Karafiát, L. Burget, J. Cernock, and S. Khudanpur, "Recurrent neural network based language model." in Interspeech, 2010, pp 1045-1048.

[7] Y. Bengio, H. Schwenk, J.-S. Senécal, F. Morin, and J.-L. Gauvain, "Neural probabilistic language models," in Innovations in Machine Learning. Springer, 2006, pp. 137 – 186.

[8] P. J. Werbos, "Backpropagation through time: What it does and how to do it," Proceedings of the IEEE, vol. 78, no. 10, pp. 1550 – 1560, 1990.

[9] J. Guo, "BackPropagation Through Time," 2013.

[10] T. Mikolov, S. Kombrink, A. Deoras, L. Burget, and J. Cernocky, "RNNLM-Recurrent neural network language modeling toolkit," in Proc. Of the 2011 ASRU Workshop, 2011, pp. 196 – 201.

[11] W. Zaremba, I. Sutskever, and O. Vinyals, "Recurrent neural network regularization," arXiv preprint arXiv:1409.2329, 2014.

[12] P. Ace, P. Schwarz, and V. P. Ace, "Phoneme recognition based on long temporal context," 2009.

[13] N. Brummer and D. A. van Leeuwen, "On calibration of language recognition scores," in Speaker and Language Recognition Workshop, 2006. IEEE Odyssey 2006: The. IEEE, 2006, pp. 1 – 8.

[14] A. F. Martin and C. S. Greenberg, "The 2009 NIST Language Recog-nition Evaluation." in Odyssey, 2010, p. 30.

[15] S. Lai, K. Liu, L. Xu, and J. Zhao, "How to Generate a Good Word Embedding?" arXiv preprint arXiv:1507.05523, 2015.

[16] Q. Le, T. Mikolov, "Distributed representations of sentences and documents" arXiv preprint arXiv:1405.4053.2014

[17] S. Yujing, X. Yeming, X. Ji, P. Jelin, Y. Yonghong, "Recurrent Neural Network language model with vector space word representations" in Proceedings of the 21th International Congress on Sound and Vibration, 2014.

[18] N. Brummer, L. Burget, J. H. Cernocky, O. Glembek, F. Grezl, M. Karafiat, D. A. V. Leeuwen, P. Matejka, P. Schwarz, and A. Strasheim, "Fusion of heterogeneous speaker recognition systems in the STBU submission for the NIST speaker recognition evaluation 2006," Audio, Speech, and Language Processing, IEEE Transactions on, vol. 15, no. 7, pp. 2072 – 2084, 2007.

[19] T. Mikolov, "Statistical language models based on neural networks," Ph.D. dissertation, Ph. D. thesis, Brno University of Technology, 2012.

[20] J. L. McClelland, D. E. Rumelhart, R. G. Pdp et al., "Parallel distributed processing: Explorations in the microstructures of cognition, volume 2: Psychological and biological models," MIT Press, vol. 76, p. 1555, 1986.

[21] J. Goodman, "Classes for fast maximum entropy training," in Acoustics, Speech, and Signal Processing, 2001. Proceedings.(ICASSP'01). 2001 IEEE International Conference on, vol. 1. IEEE, 2001, pp. 561 – 564.

[22] S. Davis and P. Mermelstein, "Comparison of parametric representations for monosyllabic word recognition in continuously spoken sentences,"
Acoustics, Speech and Signal Processing, IEEE Transactions on, vol. 28, no. 4, pp. 357 – 366, 1980.